Contribution ID: **169** Type: **not specified**

Using eBPF as a heterogeneous processing ABI

Thursday, 15 November 2018 12:00 (20 minutes)

eBPF (extended Berkeley Packet Filter) is an in-kernel generic virtual machine, which can be used to execute simple programs injected by the user at various hooks in the kernel, on the occurrences of events such as incoming packets. eBPF was designed to simplify the work of in-kernel just-in-time compilers, i.e. translation of eBPF intermediate representation to CPU machine code. Upstream Linux kernel currently contains JITs for all major 64-bit instruction set architectures (ISAs) (x86, AArch64, MIPS, PowerPC, SPARC, s390) as well as some 32-bit translators (ARM, x86, also NFP - Netronome Flow Processor).

The eBPF generic virtual machine with clearly defined semantics makes it a very good vehicle for enabling programming of custom hardware. From storage devices to networking processors most host I/O controllers today are built based on or with accompaniment with general purpose processing cores, e.g. ARM. As vendors try to expose more and more capabilities of their hardware, using a general purpose machine definition like eBPF to inject code into hardware directly allows us to avoid creation of vendor specific APIs.

In this talk I will describe the eBPF offload mechanism which exists today in the Linux kernel and how they compare to other offloading stacks e.g. for compute or graphics. I will present a proof-of-concept work on of reusing existing eBPF JITs for non-host architecture (e.g. ARM JIT on x86) to program a emulated device, followed by a short description of the eBPF offload for NFP hardware as an example of a real-life offload.

I agree to abide by the anti-harassment policy

Presenter: KICINSKI, Jakub (Netronome)

Session Classification: BPF MC